EXPRESS MAIL LABEL NO.:EJ922406444US DATE OF DEPOSIT: April 6, 2000

I hereby certify that this paper and fee are being deposited with the United States Postal Service Express Mail Post Office to Addressee service under 37 CFR §1.10 on the date indicated above and is addressed to the Assistant Commissioner of Patents, Washington, D.C. 20231.

Dianne Lane
NAME OF PERSON MAILING PAPER AND FEE

Dianne Lane
SIGNATURE OF PERSON MAILING PAPER AND FEE

INVENTORS: Arthur R. Francis and Brad B. Topol

SYSTEM, APPARATUS AND METHOD FOR TRANSFORMATION OF JAVA SERVER PAGES INTO PVC FORMATS

BACKGROUND OF THE INVENTION

1. Technical Field:

The present invention is directed to a system, apparatus and method for the transformation of Java server pages into formats for various pervasive computing (PvC) devices.

2. Description of Related Art:

The use of Java Server Pages (JSPs) for the generation of dynamic HyperText Mark-up Language (HTML) content is becoming more popular. JSP provides a methodology by which information related to the content

and information related to the presentation of the content may be separated. For example, JSP provides a <bean> tag which is utilized to obtain content, such as Java applets, while HTML tags are used to specify presentation of the content.

A problem arises in these prior JSPs in that they are designed with the assumption that the presentation of the content will be rendered in a browser that executes on a desktop workstation with a full-sized, high resolution, full-color monitor. As the use of mobile handheld pervasive computing (PvC) devices becomes more prevalent, the assumption used in the design of JSPs becomes a stumbling block since there is a need to deliver the JSP to a variety of PvC devices whose browsing screen size capabilities are drastically different from that of a desktop workstation.

One way to address this problem is to have a transcoding proxy server, or servlet, intercept the JSP and transcode it dynamically, i.e., on the fly, as the JSP is being sent to the PvC device. This option results in extra overhead costs incurred each and every time the JSP is invoked.

A second approach to solving this problem is to have a human JSP developer design and create separate JSPs for each and every PvC device one might want to support. This approach has the disadvantage of extra cost required to have the additional JSPs authored by a human developer who needs to be cognizant of the capabilities of the PvC devices he or she intends to support.

Thus, it would be advantageous to have a system, apparatus and method that provides a mechanism by which JSPs may be automatically converted to formats useable by a plurality of PvC devices.

SUMMARY OF THE INVENTION

A system, apparatus and method for transforming Java Server Pages (JSPs) into pervasive computing (PvC) device specific JSPs. An original JSP is parsed for JSP tags. The JSP tags are converted to HyperText Mark-up Language (HTML) comment tags. The JSP is then transformed into a PvC device specific JSP by converting the HTML tags in the JSP to HTML tags for a specific PvC device. The HTML comment tags are then parsed for the comment tags having embedded JSP tags. The JSP tags are then restored by removing the HTML comment tag identifiers. A PvC device specific JSP is thus created.

BRIEF DESCRIPTION OF THE DRAWINGS

The novel features believed characteristic of the invention are set forth in the appended claims. The invention itself, however, as well as a preferred mode of use, further objectives and advantages thereof, will best be understood by reference to the following detailed description of an illustrative embodiment when read in conjunction with the accompanying drawings, wherein:

Figure 1 is an exemplary block diagram of a distributed data processing system according to the present invention;

Figure 2 is an exemplary block diagram of a server according to the present invention;

Figure 3 is an exemplary diagram illustrating the use of HTML tags to create an HTML document;

Figure 4 is an exemplary diagram of the use of JSP tags along with HTML tags in a JSP;

Figure 5 is an exemplary diagram illustrating the phases of operation of the present invention;

Figure 6 is a flowchart outlining an exemplary operation of the present invention; and

Figure 7 is a flowchart outlining an exemplary operation of the present invention for converting a JSP file to a PvC specific JSP file.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

With reference now to the figures, and in particular with reference to **Figure 1**, a pictorial representation of a distributed data processing system is depicted in which the present invention may be implemented. Distributed data processing system **100** is a network of computers in which the present invention may be implemented. Distributed data processing system **100** contains network **102**, which is the medium used to provide communications links between various devices and computers connected within distributed data processing system **100**. Network **102** may include permanent connections, such as wire or fiber optic cables, or temporary connections made through telephone connections. Network **102** may further include wireless communication connections through, for example, base stations, satellites, and the like, for communicating with wireless communication devices.

In the depicted example, server **104** is connected to network **102**, along with storage unit **106**. In addition, clients **108**, **110**, **112** and **118** are also connected to network **102**. These clients, **108**, **110**, **112** and **118**, may be, for example, personal computers, network computers, personal digital assistants (PDAs), alphanumeric pagers, cellular telephones, and the like. Any communication device, either wired or wireless, that is capable of receiving and displaying a Java Server Page (JSP) is intended to be within the scope of the term "client" as used in this disclosure.

In the depicted example, server **104** provides data, such as boot files, operating system images and applications, and in particular JSPs, to clients **108-118**. Clients **108-118** are clients to server **104**. Distributed data processing system **100** may include additional servers, clients, and other devices not shown.

In the depicted example, distributed data processing system **100** is the Internet, with network **102** representing a worldwide collection of networks and gateways that use the TCP/IP suite of protocols to communicate with one another. At the heart of the Internet is a backbone of high-speed data communication lines

between major nodes or host computers consisting of thousands of commercial, government, education, and other computer systems that route data and messages. Of course, distributed data processing system **100** also may be implemented as a number of different types of networks such as, for example, an intranet or a local area network. **Figure 1** is intended as an example and not as an architectural limitation for the processes of the present invention.

Referring to **Figure 2**, a block diagram of a data processing system which may be implemented as a server, such as server **104** in **Figure 1**, is depicted in accordance with the present invention. Data processing system **200** may be a symmetric multiprocessor (SMP) system including a plurality of processors **202** and **204** connected to system bus **206**. Alternatively, a single processor system may be employed. Also connected to system bus **206** is memory controller/cache **208**, which provides an interface to local memory **209**. I/O bus bridge **210** is connected to system bus **206** and provides an interface to I/O bus **212**. Memory controller/cache **208** and I/O bus bridge **210** may be integrated as depicted. Peripheral component interconnect (PCI) bus bridge **214** connected to I/O bus **212** provides an interface to PCI local bus **216**. A number of modems **218-220** may be connected to PCI bus **216**. Typical PCI bus implementations will support four PCI expansion slots or add-in connectors.

Communications links to network computers **108-112** in **Figure 1** may be provided through modem **218** and network adapter **220** connected to PCI local bus **216** through add-in boards. Additional PCI bus bridges **222** and **224** provide interfaces for additional PCI buses **226** and **228**, from which additional modems or network adapters may be supported. In this manner, server **200** allows connections to multiple network computers. A memory mapped graphics adapter **230** and hard disk **232** may also be connected to I/O bus **212** as depicted, either directly or indirectly.

Those of ordinary skill in the art will appreciate that the hardware depicted in **Figure 2** may vary. For example, other peripheral devices, such as optical disk drives and the like, also may be used in addition to or in place of the hardware depicted. The depicted example is not meant to imply architectural limitations with respect to the

present invention. The data processing system depicted in **Figure 2** may be, for example, an IBM RISC/System 6000, a product of International Business Machines Corporation in Armonk, New York, running the Advanced Interactive Executive (AIX) operating system.

The server **200** stores a Java Server Page (JSP) that is designed for presentation on a typical desktop workstation, such as client device **108**. The JSP is comprised of, for example, text, images and tags that are used to generate a JSP presentation on a client device when the JSP is downloaded to the client device. For example, the JSP may comprise text, HyperText Mark-up Language (HTML) tags identifying presentation formats of the text, HTML tags identifying in-line image files to be displayed, and JSP tags. JSP tags are used to encapsulate scripting language fragments into an HTML page. JSP pages use JSP tags or scriptlets to generate the dynamic content on the page (the content that changes according to the request, such as requested account information or the price of a specific bottle of wine). The logic that generates the content is encapsulated in tags and JavaBean components, and tied together in scriptlets, all of which are executed on the server side. More information regarding JSP technology and JSP tags may be obtained from the JSP section of the Sun Corporation Web site at http://java.sun.com/products/jsp, which is hereby incorporated by reference.

While the present invention will be described with reference to HTML tags, one of ordinary skill in the art should appreciate that any mark-up language, such as XML, may be utilized without departing from the spirit and scope of the present invention.

Figure 3 illustrates an example of the use of HTML tags to dictate the manner in which text information is displayed by a browser on a client device. As shown in **Figure 3**, the HTML tags may include tags identifying headers, titles, text bodies, paragraphs, underlining, bold text, and the like. The bottom portion of **Figure 3** shows how these tags are processed by the browser application to generate a formatted Web page for display to a user.

JSPs use tags in a similar manner as the HTML tags shown in **Figure 3** to designate the format of the JSP page as it will be displayed. The JSPs further include JSP tags designating JSP components, such as Java beans or applets, that are to be executed when accessing the JSP. **Figure 4** provides an example of the use of HTML tags and JSP tags for a simplified JSP. As shown in **Figure 4**, the JSP includes HTML tags **410** and **420** that designate formatting of text information **430**. The JSP further includes JSP tag **440**, which reads "<jsp:forward page="{www.hauntedhouse.com|<%=HauntedHouse.class%>}"/>, that forwards a client request to a JSP file "HauntedHouse.class" for processing.

The JSP tags are applicable to any client device. Thus, the JSP tags need not be modified when converting a general JSP to a PvC device specific JSP for a PvC device having different capabilities than a desktop workstation. The HTML tags, however, do require modification so that they are consistent with the capabilities of the PvC device to which the JSP is transmitted.

The present invention provides a system, apparatus and method for converting JSPs designed for desktop workstations, into JSPs that are useable by a plurality of PvC devices. The present invention parses a JSP to locate and mask the JSP tags and convert the HTML tags to PvC device specific HTML tags. The method according to the present invention may be implemented, for example, as one or more Java applications, applets, or Java beans resident on a server of a distributed data processing system, such as server **140** or server **200**.

The Java beans may be executed when, for example, a system operator designates, via an input device (not shown), that a JSP is to be converted for use by PvC devices. Alternatively, the Java beans may be executed when a request is first received from a PvC device.

For example, with reference to **Figures 1** and **2**, when a request for a JSP is received by a server **200** from a PvC device, such as PvC device **110**, the server **200** receives the request via the network adapter **220**. The server **200** then processes request by parsing header information of the request to determine the type of PvC

device from which the request was received. For example, the header of the request may include a device type identifier indicating the type of device that sent the request.

The server **200** may then search for a stored version of the requested JSP on hard disk **232** corresponding to the PvC device from which the request was received. This search may be based on, for example, filename and filename extension. The various PvC device specific JSPs may be stored under the same filename with different extensions designating the PvC for which the JSP is formatted. Alternatively, the filenames may be different based on the PvC for which they are formatted.

If an appropriate version is available, the JSP is sent to the PvC device. However, if an appropriate version is not available, the server **200** may invoke the Java beans to convert the JSP into a PvC specific JSP for the particular PvC device sending the request. This PvC specific JSP will then be stored on the server and utilized when later requests from similar PvC devices are received.

Figure 5 is an illustration of the various phases implemented by the present invention. As shown in **Figure 5**, phase I of operating involves parsing the original JSP file **510** to produce a resultant file **520**. Through parsing of the original JSP file **510**, the HTML tags and JSP tags in the JSP are identified. The HTML tags and JSP tags are identified by looking for the occurrence of symbols designating the various tags. As the JSP file **510** is parsed, HTML constructs and HTML tags are written to a resultant file **520**. JSP tags are masked by writing them to the resultant file **520** as HTML comment tags. Thus, the resultant file **520** contains only valid HTML constructs, i.e., HTML tags and textual information. HTML tags are indicators in HTML documents that identify document elements, structure, formatting, and hypertext links to other documents or to included media. Enclosed within these tags is textual information that is presented to the display device.

Next, in phase II of the operation, the resultant file **520** is transcoded into one or more PvC device specific files **530**. The JSP tags that are embedded as HTML comment tags remain in the transcoded files **530** as HTML comment tags. The

transcoding of the resultant file **520** into PvC specific files **530** involves placing the identified HTML tags into a document object model and performing a transcoding node by node. As is know to those of ordinary skill in the art, a document object model provides a representation of the hierarchical arrangement of document tags in a tree-like structure. Each node of the document object model represents a document tag.

The transcoding of the present invention proceeds through the document object model node by node and converts the HTML tag associated with the node to a corresponding tag in a mark-up language useable by the particular PvC device. For example, the HTML tag may be converted to an equivalent Wireless Mark-up Language (WML) tag or a Handheld Device Markup Language (HDML) tag. Any tags that aren't supported by the mark-up language for the PvC device are removed and replaced with a closest equivalent.

As a part of this processing phase, the occurrence of in-line images will also be transcoded to the resolution of the target device. As a result of the image transcoding process, static representation of the original image will be saved with a new file extension that indicates the image conversion format. In conjunction with the image transcoding process, the links to the images in each of the generated files will be updated to reflect the transcoded file that contains the resolution requirements of the target PvC device.

In the third phase of the operation, the JSP tags are unmasked. This phase involves parsing the transcoded files, locating the hidden JSP tags and converting the file back to a valid JSP file. This may be done by parsing the transcoded files, identifying HTML comment tags, determining if the content of the HTML comment tags fit the format of a JSP tag, and removing the HTML comment tag identifiers when saving the transcoded file to thereby restore the JSP tags. In this way, a JSP file is generated that is transcoded for a specific PvC device.

Figure 6 is a flowchart outlining an exemplary operation of the present invention. As shown in **Figure 6**, the operation starts with receiving a request for a

JSP from a PvC device (step **601**). A search is then conducted for a version of the JSP that is useable by the requesting PvC device (step **602**). A determination is made as to whether or not a JSP for the requesting PvC device is found (step **603**). If so, the JSP is sent to the PvC device (step **604**). If not, a JSP for the requesting PvC device is generated and stored (step **605**). The generated JSP is then sent to the requesting PvC device (step **606**). The operation then ends.

While step **605** above is described as generating only the PvC specific JSP for the requesting PvC device, the operation is not limited to such. Rather, the PvC specific JSPs for all supported PvC devices may be generated. Additionally, the PvC specific JSPs may be generated in response to a command received from a user, such as a system operator.

Figure 7 is a flowchart outlining an exemplary operation of the present invention when generating a PvC specific JSP from an original JSP. This operation may be implemented as step **605** in **Figure 6**, described above.

As shown in **Figure 7**, the operation starts with reading a JSP file and storing the contents of the JSP file in a buffer memory (step **701**). The contents of the JSP file are then parsed (step **702**). A determination is made as to whether or not a tag is encountered (step **703**). If not, the contents are written to a resultant file (step **704**). If so, a determination is made as to whether the tag is an HTML tag (step **705**). If so, the HTML tag is written to the resultant file (step **706**).

If not, the tag must be a JSP tag. The JSP tag is then converted to an HTML comment tag and written to the resultant file (step **707**). The process is then repeated until an end of file is encountered (step **708**).

Thereafter, it is determined for which PvC devices the JSP will be converted (step **709**). Then, for each PvC device, the following steps are performed. The HTML tags in the resultant file are transcoded to PvC device specific tags (step **710**). A transcoded file is then stored for the PvC device containing the transcoded HTML tags and resultant file content (step **711**). A determination is then made as to whether

there are more PvC devices for which the JSP should be transcoded (step **712**). If so, the operation returns to step **710**. If not, the operation proceeds to phase III.

In phase III, the operation performs the following steps for each of the PvC specific transcoded files that were created. The transcoded file is parsed for HTML comments tags (step **713**). A determination is made as to whether a tag is encountered (step **714**). If not, the operation continues to parse the transcoded file until an end of file is encountered (step **718**) at which time the operation ends.

If a tag is encountered, it is determined whether or not the tag is an HTML comment tag (step **715**). If not, the operation continues to parse the transcoded file until an end of file is encountered (step **718**) at which time the operation ends. If so, a determination is made as to whether the HTML comment tag contains a JSP tag (step **716**). If not, the operation goes to step **718**. If so, the operation removes the HTML comment tag identifiers (step **717**) and continues to parse the transcoded file until an end of file is encountered (step **718**). The transcoded file with the restored JSP tags is then stored (step **719**). After all of the transcoded files are processed, the operation ends.

Thus, the present invention provides a system, apparatus and method for automatically generating PvC specific JSP files from a general JSP file. In this way, the expense of providing JSP files for a variety of PvC devices is greatly reduced.

It is important to note that while the present invention has been described in the context of a fully functioning data processing system, those of ordinary skill in the art will appreciate that the processes of the present invention are capable of being distributed in the form of a computer readable medium of instructions and a variety of forms and that the present invention applies equally regardless of the particular type of signal bearing media actually used to carry out the distribution. Examples of computer readable media include recordable-type media such a floppy disc, a hard disk drive, a RAM, CD-ROMs, and transmission-type media such as digital and analog communications links.

The description of the present invention has been presented for purposes of illustration and description, and is not intended to be exhaustive or limited to the invention in the form disclosed. Many modifications and variations will be apparent to those of ordinary skill in the art. The embodiment was chosen and described in order to best explain the principles of the invention, the practical application, and to enable others of ordinary skill in the art to understand the invention for various embodiments with various modifications as are suited to the particular use contemplated.